SuperPrice Application Testing Documentation

Nathan Nguyen

Overview:

This document outlines the test cases for the Superprice application. The test focuses on various features like filtering by supermarket, comparing prices across stores, searching by product name, and sorting by price.

Setup:

A mock repository, SupermarketItemRepository, is created to simulate interactions with the database.

Three mock supermarket items are defined for testing.

Before each test, the repository behavior is set up to return specific items based on different conditions.

Tests:

Filter by Supermarket (US1.1)

Goal: Check if items can be filtered by their supermarket.

Test: Fetch items from "SupermarketA" and ensure only the right items are returned.

Compare Price Across Supermarkets (US1.2)

Goal: Compare the price of a product from different supermarkets.

Test: Search for "Item1" and check if the correct price is returned.

Search by Product Name (US2.1)

Goal: Find products using their name.

Test: Search for "Item1" and ensure the right product is returned.

Filter by Price Range (US4.1)

Goal: Filter items based on a price range.

Test: Fetch items priced between 100.0 and 200.0 and ensure the right number of items are returned.

Sort by Price (US5.1)

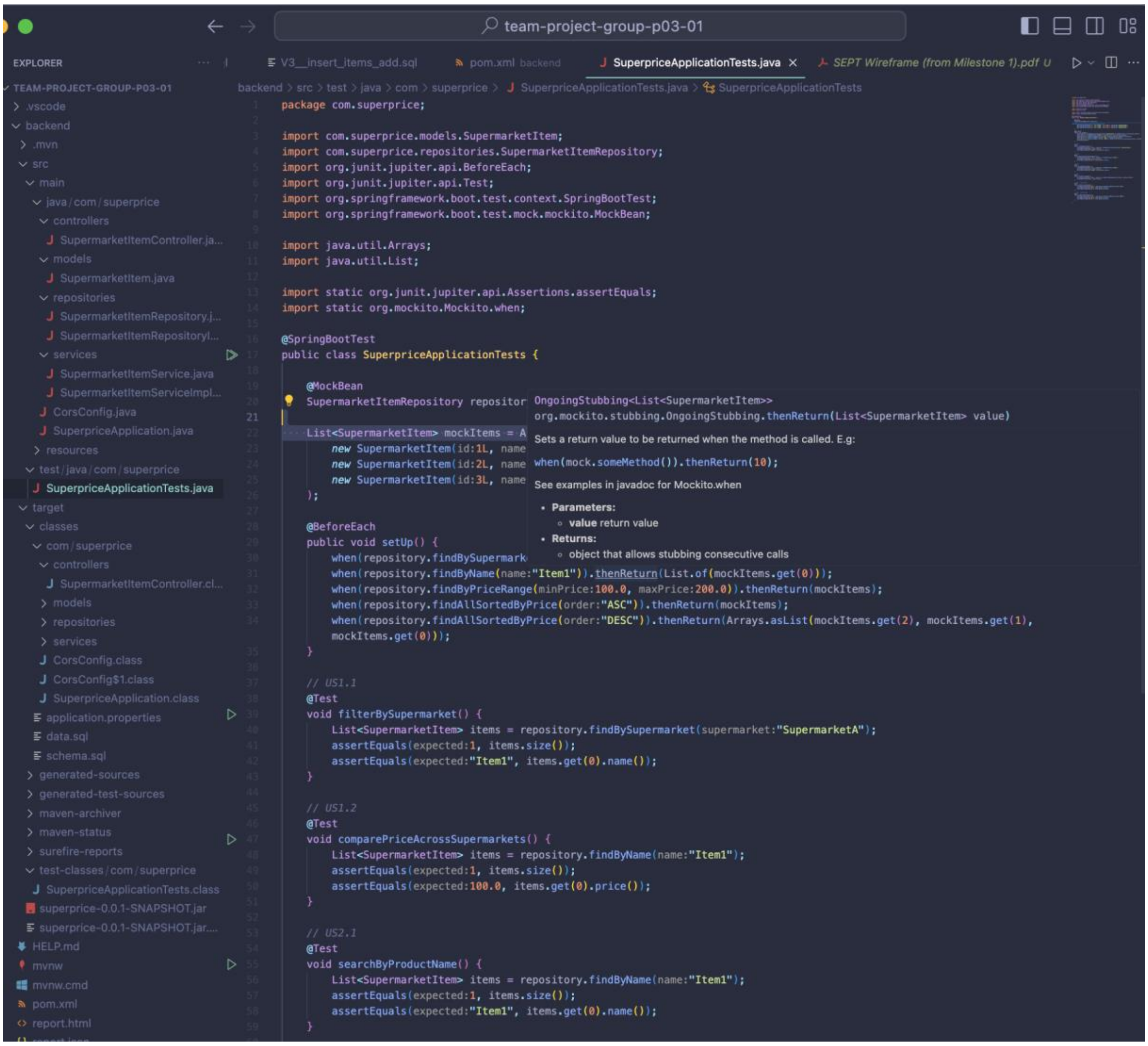
Goal: Sort items based on their price in both ascending and descending order.

Tests:

For ascending, ensure the items are in the correct order from lowest to highest price.

For descending, ensure the items are in the correct order from highest to lowest price

Screenshots of test and successful building are below:



team-project-group-p03-01
EXPLORER
V3__insert_items_add.sql
pom.xml backend
SuperpriceApplicationTests.java
SEPT Wireframe (from Milestone 1).pdf U
backend > src > test > java > com > superprice > SuperpriceApplicationTests.java > SuperpriceApplicationTests
TEAM-PROJECT-GROUP-P03-01
.vscode
backend
.mvn
src
main
java / com / superprice
controllers
SupermarketItemController.ja...
models
SupermarketItem.java
repositories
SupermarketItemRepository.j...
SupermarketItemRepositoryI...
services
SupermarketItemService.java
SupermarketItemServiceImpl...
CorsConfig.java
SuperpriceApplication.java
resources
test / java / com / superprice
SuperpriceApplicationTests.java
target
classes
com / superprice
controllers
SupermarketItemController.cl...
models
repositories
services
CorsConfig.class
CorsConfig$1.class
SuperpriceApplication.class
application.properties
data.sql
schema.sql
generated-sources
generated-test-sources
maven-archiver
maven-status
surefire-reports
test-classes / com / superprice
SuperpriceApplicationTests.class
superprice-0.0.1-SNAPSHOT.jar
superprice-0.0.1-SNAPSHOT.jar....
HELP.md
mvnw
mvnw.cmd
pom.xml
report.html

```java
package com.superprice;

import com.superprice.models.SupermarketItem;
import com.superprice.repositories.SupermarketItemRepository;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.mock.mockito.MockBean;

import java.util.Arrays;
import java.util.List;

import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.mockito.Mockito.when;

@SpringBootTest
public class SuperpriceApplicationTests {

    @MockBean
    SupermarketItemRepository repositor

    List<SupermarketItem> mockItems = A
        new SupermarketItem(id:1L, name
        new SupermarketItem(id:2L, name
        new SupermarketItem(id:3L, name
    );

    @BeforeEach
    public void setUp() {
        when(repository.findBySupermark
        when(repository.findByName(name:"Item1")).thenReturn(List.of(mockItems.get(0)));
        when(repository.findByPriceRange(minPrice:100.0, maxPrice:200.0)).thenReturn(mockItems);
        when(repository.findAllSortedByPrice(order:"ASC")).thenReturn(mockItems);
        when(repository.findAllSortedByPrice(order:"DESC")).thenReturn(Arrays.asList(mockItems.get(2), mockItems.get(1),
        mockItems.get(0)));
    }

    // US1.1
    @Test
    void filterBySupermarket() {
        List<SupermarketItem> items = repository.findBySupermarket(supermarket:"SupermarketA");
        assertEquals(expected:1, items.size());
        assertEquals(expected:"Item1", items.get(0).name());
    }

    // US1.2
    @Test
    void comparePriceAcrossSupermarkets() {
        List<SupermarketItem> items = repository.findByName(name:"Item1");
        assertEquals(expected:1, items.size());
        assertEquals(expected:100.0, items.get(0).price());
    }

    // US2.1
    @Test
    void searchByProductName() {
        List<SupermarketItem> items = repository.findByName(name:"Item1");
        assertEquals(expected:1, items.size());
        assertEquals(expected:"Item1", items.get(0).name());
    }
```

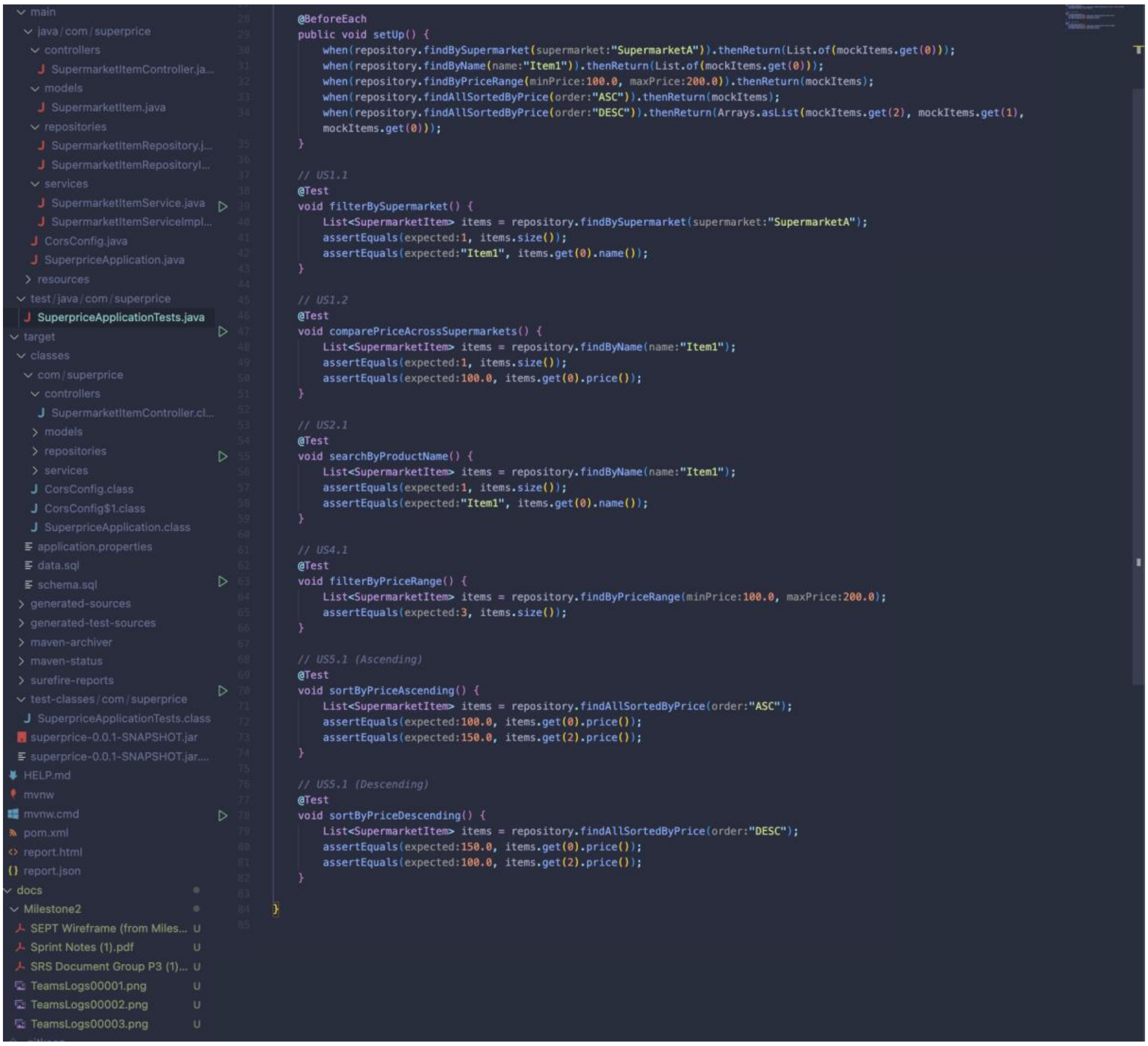
```
OngoingStubbing<List<SupermarketItem>>
org.mockito.stubbing.OngoingStubbing.thenReturn(List<SupermarketItem> value)
```

Sets a return value to be returned when the method is called. E.g:

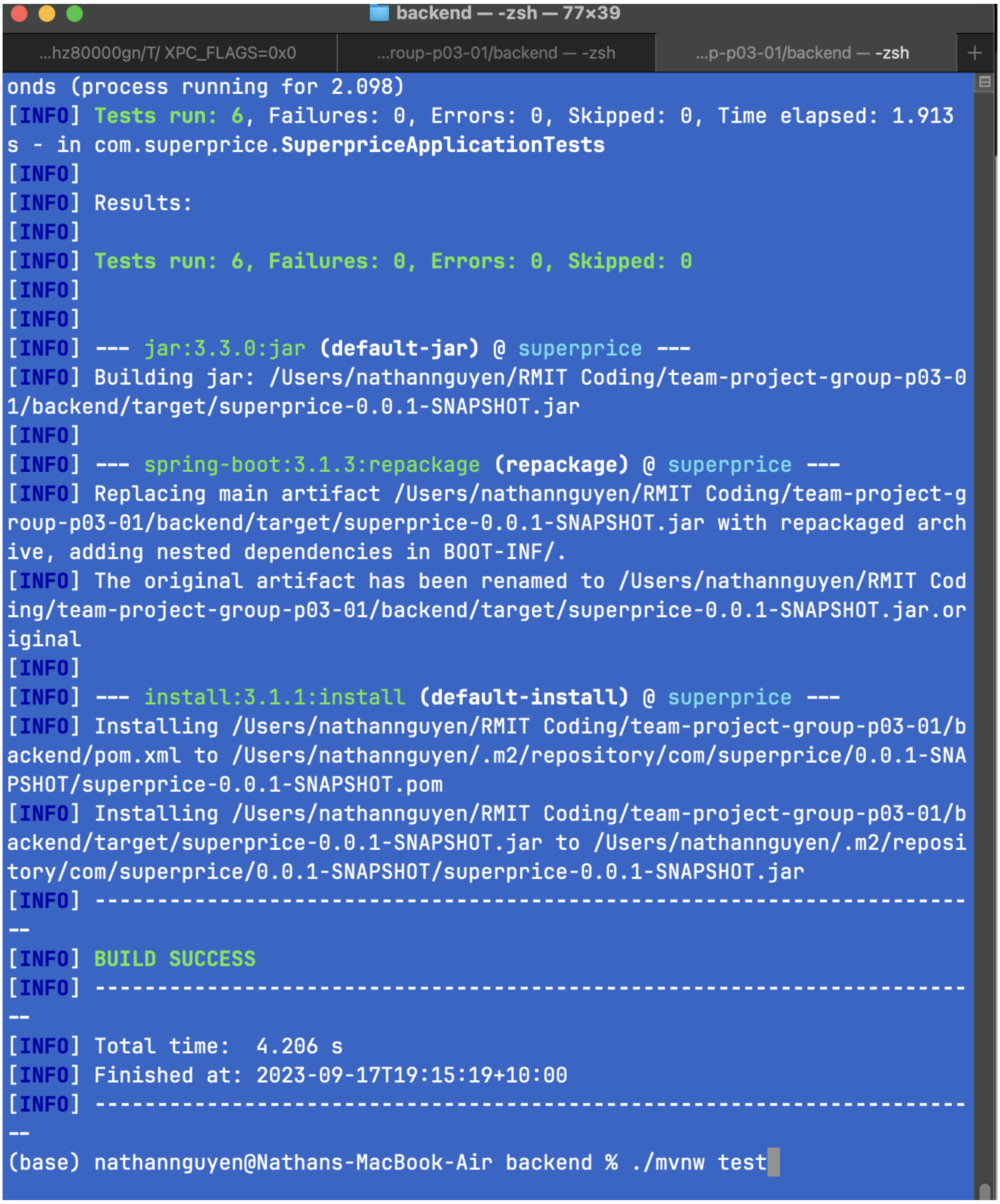
`when(mock.someMethod()).thenReturn(10);`

See examples in javadoc for Mockito.when

- **Parameters:**
  - **value** return value
- **Returns:**
  - object that allows stubbing consecutive calls


main
java / com / superprice
controllers
SupermarketItemController.ja...
models
SupermarketItem.java
repositories
SupermarketItemRepository.j...
SupermarketItemRepositoryI...
services
SupermarketItemService.java
SupermarketItemServiceImpl...
CorsConfig.java
SuperpriceApplication.java
resources
test / java / com / superprice
SuperpriceApplicationTests.java
target
classes
com / superprice
controllers
SupermarketItemController.cl...
models
repositories
services
CorsConfig.class
CorsConfig$1.class
SuperpriceApplication.class
application.properties
data.sql
schema.sql
generated-sources
generated-test-sources
maven-archiver
maven-status
surefire-reports
test-classes / com / superprice
SuperpriceApplicationTests.class
superprice-0.0.1-SNAPSHOT.jar
superprice-0.0.1-SNAPSHOT.jar....
HELP.md
mvnw
mvnw.cmd
pom.xml
report.html
report.json
docs
Milestone2
SEPT Wireframe (from Miles... U
Sprint Notes (1).pdf U
SRS Document Group P3 (1)... U
TeamsLogs00001.png U
TeamsLogs00002.png U
TeamsLogs00003.png U

```java
    @BeforeEach
    public void setUp() {
        when(repository.findBySupermarket(supermarket:"SupermarketA")).thenReturn(List.of(mockItems.get(0)));
        when(repository.findByName(name:"Item1")).thenReturn(List.of(mockItems.get(0)));
        when(repository.findByPriceRange(minPrice:100.0, maxPrice:200.0)).thenReturn(mockItems);
        when(repository.findAllSortedByPrice(order:"ASC")).thenReturn(mockItems);
        when(repository.findAllSortedByPrice(order:"DESC")).thenReturn(Arrays.asList(mockItems.get(2), mockItems.get(1),
        mockItems.get(0)));
    }

    // US1.1
    @Test
    void filterBySupermarket() {
        List<SupermarketItem> items = repository.findBySupermarket(supermarket:"SupermarketA");
        assertEquals(expected:1, items.size());
        assertEquals(expected:"Item1", items.get(0).name());
    }

    // US1.2
    @Test
    void comparePriceAcrossSupermarkets() {
        List<SupermarketItem> items = repository.findByName(name:"Item1");
        assertEquals(expected:1, items.size());
        assertEquals(expected:100.0, items.get(0).price());
    }

    // US2.1
    @Test
    void searchByProductName() {
        List<SupermarketItem> items = repository.findByName(name:"Item1");
        assertEquals(expected:1, items.size());
        assertEquals(expected:"Item1", items.get(0).name());
    }

    // US4.1
    @Test
    void filterByPriceRange() {
        List<SupermarketItem> items = repository.findByPriceRange(minPrice:100.0, maxPrice:200.0);
        assertEquals(expected:3, items.size());
    }

    // US5.1 (Ascending)
    @Test
    void sortByPriceAscending() {
        List<SupermarketItem> items = repository.findAllSortedByPrice(order:"ASC");
        assertEquals(expected:100.0, items.get(0).price());
        assertEquals(expected:150.0, items.get(2).price());
    }

    // US5.1 (Descending)
    @Test
    void sortByPriceDescending() {
        List<SupermarketItem> items = repository.findAllSortedByPrice(order:"DESC");
        assertEquals(expected:150.0, items.get(0).price());
        assertEquals(expected:100.0, items.get(2).price());
    }

}
```


backend — -zsh — 77×39
...hz80000gn/T/ XPC_FLAGS=0x0
...roup-p03-01/backend — -zsh
...p-p03-01/backend — -zsh

```
onds (process running for 2.098)
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.913
s - in com.superprice.SuperpriceApplicationTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jar:3.3.0:jar (default-jar) @ superprice ---
[INFO] Building jar: /Users/nathannguyen/RMIT Coding/team-project-group-p03-0
1/backend/target/superprice-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot:3.1.3:repackage (repackage) @ superprice ---
[INFO] Replacing main artifact /Users/nathannguyen/RMIT Coding/team-project-g
roup-p03-01/backend/target/superprice-0.0.1-SNAPSHOT.jar with repackaged arch
ive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to /Users/nathannguyen/RMIT Cod
ing/team-project-group-p03-01/backend/target/superprice-0.0.1-SNAPSHOT.jar.or
iginal
[INFO]
[INFO] --- install:3.1.1:install (default-install) @ superprice ---
[INFO] Installing /Users/nathannguyen/RMIT Coding/team-project-group-p03-01/b
ackend/pom.xml to /Users/nathannguyen/.m2/repository/com/superprice/0.0.1-SNA
PSHOT/superprice-0.0.1-SNAPSHOT.pom
[INFO] Installing /Users/nathannguyen/RMIT Coding/team-project-group-p03-01/b
ackend/target/superprice-0.0.1-SNAPSHOT.jar to /Users/nathannguyen/.m2/reposi
tory/com/superprice/0.0.1-SNAPSHOT/superprice-0.0.1-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
--
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
--
[INFO] Total time:  4.206 s
[INFO] Finished at: 2023-09-17T19:15:19+10:00
[INFO] ------------------------------------------------------------------------
--
(base) nathannguyen@Nathans-MacBook-Air backend % ./mvnw test
```